Guía día 2

Ejercicio 1: Crear un filtro para CCAA (dropdown)

- Copiar ejercicio 4 completo día 1
- Crear carpeta “utils” y “selector_ccaa.py”
- Imports:

```
- import sys
- import os
- sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file
  __), '..')))
-
- from dash import dcc
- import dash_bootstrap_components as dbc
- import pandas as pd
-
- from utils import functions as f # type: ignore
```

- Lista de CCAAS y diccionario de opciones

```
    ccaas = f.consulta("""SELECT distinct Id, Descripcion FROM
[SGP_SIPE].[cat].[CCAA]""")

    ccaas = ccaas.sort_values(by= "Descripcion")

    options_dict = [{'label': ccaa["Descripcion"], 'value': ccaa["Id"]}
for _, ccaa in ccaas.iterrows() if ccaa["Descripcion"] != "CASTILLA Y
LEÓN" and ccaa["Descripcion"] != "COMUNIDAD DE MADRID" and
ccaa["Descripcion"] != "MELILLA"]
```

- Crear Selector

```
    selector =  dbc.Card([
                    dbc.CardBody(dcc.Dropdown(
                                id = id,
                                options= options_dict,
                                clearable=False,
                                className="app-stock-selector",
                                placeholder= "Selecciona o introduce
Comunidad Autónoma",
                                style = {"font-size": "15px","font-
family": "Arial, sans-serif", "maxHeight": "500px"}))],
                                className="mt4",
```

```
                                style = {"border": "none",
                                    "borderRadius": "0",
                                    "width": "100%",
                                    "height": "70px",
                                    "boxShadow": "none"})

    return selector
```

- Cambiar la estructura del layout
    - Cambiar título a menú

```
menu = dbc.Row([
    dbc.Col(html.H1("Ventas y Desembarques por CCAA", style={"textAlign":
"center", "color": "#000000", "font-size": 40}), width = 9),
   dbc.Col(se.selector_ccaa(),
width=3)
])
```

```
app.layout = dcc.Loading(type= "circle", fullscreen= True, children=
[menu, contenido])
```

- Añadir a contenido id = “contenido”

```
contenido = html.Div(dbc.Row(id= "contenido"))
```

- Crear callback

```
from dash.dependencies import Input, Output
```

```
@app.callback(
    Output("contenido", "children"),
    Input('selector-ccaa', 'value'),
)
def update_content(value):
    if value is None:
        return ""
```

- Copiar el contenido bajo el callback
- Adaptar las funciones para que filtren por comunidad
- Abrir los modulos y añadir value a las funciones

EJERCICIO 2: INTERACTIVIDAD ENTRE COMPONENTES

- En tabla_desembarques cambiar a PuertoBase, cambiar titulo tarjeta a desembarques por Puerto Base

```
data = data.groupby(["PuertoBase"])[["Peso",
"valor"]].sum().reset_index()
```

- Grafico provincias cambiar a idccaa_base

```
data = data[data["idccaa_base"] == value]
    data =
data.groupby("ProvinciaDesembarque")["valor"].sum().reset_index()
```

- Gráfico especies cambiar consulta a desembarques e idccaa_base

```
    data= f.consulta("SELECT * FROM SGP_CUADROSMANDO.cm.ccaa_desembarques
WHERE año = 2024")

    data = data[data["idccaa_base"] == value]
    data = data.groupby(['Especie'])[["Peso",
"valor"]].sum().reset_index().sort_values(by= "Peso", ascending = False)

    total = data["Peso"].sum()
    data["porcentaje"] = data["Peso"] / total * 100
```

- Ajustar la tabla para que se pueda seleccionar el puerto

```
                                page_size=12,
                                row_selectable = "single"
                                 )
```

- Eliminar los IDs de los módulos
- En el callback Update_content cambiar las funciones de los gráficos por IDs

```
            dbc.Col(dbc.Card([
                dbc.CardHeader(html.H4("Principales Provincias de
Desembarque", style= {"text-align": "center"}), style=
{"backgroundColor": "#f9feff"}),
                dbc.CardBody(id="barras-provincias")
                ]), width= 5),
        ]),
        dbc.Row([
```

```
            dbc.Card([dbc.CardHeader(html.H4("Variación del Número de
Buques", style= {"text-align": "center"}), style= {"backgroundColor":
"#f9feff"}),
                    dbc.CardBody(id="line-buques")
                    ])

            dbc.Col(dbc.Card([
                dbc.CardHeader(html.H4("Principales Especies", style=
{"text-align": "center"}), style= {"backgroundColor": "#f9feff"}),
                dbc.CardBody(id="pie-especies")
                ]), width= 3),
```

- Importar State y exceptions

```
-   from dash import Dash, html, dcc, exceptions
-   import dash_bootstrap_components as dbc
-   import pandas as pd
-   from dash.dependencies import Input, Output, State
```

- Crear Callback de actualizar gráficos a partir de la tabla-desembarques

```
@app.callback(
    Output("pie-especies", "children"),
    Output("barras-provincias", "children"),
    Output("line-buques", "children"),
    Input("selector-ccaa", "value"),
    Input("tabla-desembarques", "selected_rows"),
    State("tabla-desembarques", "data")
)
```

- Crear función de update_graphs

```
def update_graphs(ccaa, selected_rows, table_data):
    if ccaa is None:
        raise exceptions.PreventUpdate

    # Caso 1: no se ha seleccionado ninguna fila -> puerto=None
    if not selected_rows or len(selected_rows) == 0:
        puerto = None
    else:
        # Caso 2: fila seleccionada -> puerto correspondiente
        row_index = selected_rows[0]
        puerto = table_data[row_index]["PuertoBase"]
```

```
    # Generar gráficos
    especies = ge.graf_especies(ccaa, puerto)
    provincias = gp.provincias_desembarque(ccaa, puerto)
    buques = gb.variacion_buques(ccaa, puerto)

    return especies, provincias, buques
```

EJERCICIO PARA ALUMNOS:

AÑADIR UN FILTRO DE AÑO